Bachelor's Thesis Computing Science

Solving Robust Reachability Using Quantified Boolean Formulas

Jorrit de Boer
s1026441

July 6, 2023

First supervisor/assessor:
Dr. Sebastian Junges

Second Supervisor:
Marck van der Vegt, MSc.

Second assessor:
Dr. Nils Jansen

Radboud University

Abstract

Markov Decision Processes (MDPs) are state machines which are employed to model decision-making under uncertainty. We consider Multiple Environment Markov Decision Processes (MEMDPs) which consist of multiple MDPs that each represent a possible real world scenario. Our goal is to construct a strategy for an agent, who is unaware of which environment it is in, to reach a goal state. We provide an efficient translation from a MEMDP to a Quantified Boolean Formula (QBF formula) to allow solving the decision problem with a theorem solver. We have implemented this approach in a prototype on top of the Z3 theorem prover. The empirical evaluation shows that the translation yields large and challenging formulas. While solving MEMDPs using this translation is not yet competitive, we provide an extensive discussion of future directions that highlight the probability to obtain smaller QBF formulas.

Contents

Chapter 1

Introduction

Markov Decision Processes

Markov Decision Processes (MDPs) [2, 4, 6] are the standard framework for modeling and analyzing real-world scenarios involving both probabilistic and nondeterministic phenomena. MDPs model decision making with uncertain outcomes: in every state an agent picks an action from a set of actions and then transitions to a new state. The outcome of an action can be random: the next state the agent transitions to is governed by a probability distribution. The combination of nondeterminism (picking an action) and randomization (the transition to a new state as a result of this action) make MDPs such a powerful and widespread formalism.

To illustrate, consider the game of Rock Paper Scissors, which can be modeled as a simple MDP. The agent picks an action from the set {`rock`, `paper`, `scissors`}. The probability distribution over possible states as a result of this action represent the opponent's move. For example, this distribution could be the uniform distribution: each move has a chance of $\frac{1}{3}$. The transition to a new state reflects the game's outcome, such as a win, loss, or draw, determined by the opponent's move's probability distribution.

Consider another example where an MDP represents a medical treatment process [4]. In this context, the actions available to medical practitioners include prescribing medications and performing surgical procedures. However, the effectiveness and outcomes of these interventions are subject to uncertainty. This uncertainty arises from the unpredictable responses and potential complications associated with the treatments being administered.

In this thesis we consider *almost-sure reachability*. We want to construct a *strategy*, also called *policy*, that decides what actions to take in each state. The goal is to reach a goal state with this policy. When applying a policy to an MDP, we get certain probabilities to reach each state. In the Rock

Paper Scissors example, reaching the goal state (winning) has probability $\frac{1}{3}$, independent of the chosen strategy. For almost-sure reachability we want this chance to equal one.

Even though MDPs are widely used for modeling and analyzing dynamic systems, they have certain limitations when it comes to capturing the complexity of real-world scenarios. One key limitation is that MDPs assume a single fixed scenario, disregarding the possibility of multiple possible scenarios or variations in the underlying system. In an MDP, the agent is aware of the exact probability distribution that results from an action. In reality, however, often we are not aware of the precise probabilities that result from each action. Instead, there are various possible scenarios, each of which could induce a different distribution for each action. In these scenarios, even though we do not know precisely the MDP in which we are acting, we still want to obtain a policy to reach our goal.

To illustrate this limitation, consider another example involving a game of Rock Paper Scissors. In this game, the agent is playing against a secret opponent who could be one of three individuals: one who tends to choose rock more often, one who favors paper, and one who frequently plays scissors. The probabilities associated with each action (the agent's move) now depend on the real-world scenario, specifically on which opponent the agent is facing. This introduces a level of complexity that cannot be adequately captured by a single MDP.

Multiple Environment Markov Decision Processes

To fix this limitation, we introduce *Multiple Environment Markov Decision Processes* (MEMDPs) [2, 4, 6], the main focus of this thesis. MEMDPs are an extension of MDPs that provide a natural and even more powerful way of modeling real-world scenarios. A MEMDP consists of multiple MDPs, which we call *environments*. Each environment represents a different possible scenario. An agent still wants to reach the goal, but now has to do so without a priori knowledge about which specific environment it is in. They do this the same way as in an MDP: in each state an agent chooses one action that leads them with some probability to another state. Because the agent does not know which of the possible environments it is in, it must either take actions that progress towards the goal in all environments, or take actions to try to determine in which environment it is in. Subsequently it can then take actions to reach the goal in that specific MDP.

In the context of the MDP that models a medical treatment, consider an extension where different environments represent different patient groups that cannot be diagnosed to be in a specific group. A doctor will not know to which group a patient belongs (in what environment they are in) and

will thus have to take actions (prescribe medicines or perform surgery) that have the biggest chance of positive results for all of the possible patient groups (environments). Similarly, for a robot, different environments can model different scenarios a robot can be in, like the presence of unlocated obstacles. The robot will need to either take actions that avoid all obstacles, or take actions to locate the obstacles, which reduces the number of possible environments, after which the robot can more effectively reach their goal.

We extend almost-sure reachability to MEMDPs. Almost-sure reachability is achieved in a MEMDP when the application of a policy guarantees reaching the goal state with a probability of one across all environments. Such a policy *robustly* satisfies almost-sure reachability for the MEMDP.

Contributions

This thesis contributes a translation from the decision problem for finding a policy in a MEMDP to a Quantified Boolean Formula, see Chapter 4. This translation allows solving the decision problem with a theorem solver. We implement this approach and test its performance, see Chapter 5. Finally, we provide a discussion on how to improve the translation and its implementation to speed up solving the decision problem using this approach, see Chapter 6.

Chapter 2

Preliminaries

In this chapter we give some formal definitions which we will use in subsequent chapters. We start with MDPs and related concepts, and then do the same for MEMDPs. We give some concrete examples to provide some intuition about the concepts.

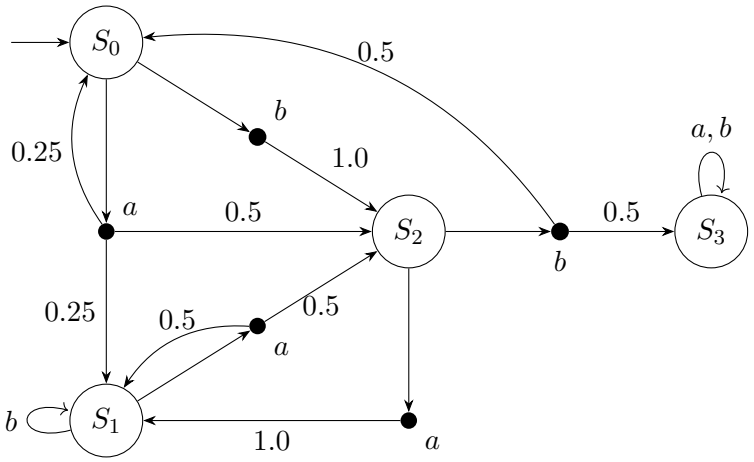
Definition 1 (MDP)**.** A Markov Decision Process (MDP) is a tuple $\mathcal{M} = \langle S, A, s_0, p \rangle$ consisting of a finite set of *states* S, a finite set of *actions* A, s_0 the *initial state*, and $p \colon S \times A \to Dist(S)$ the *transition function*.

$Dist(S)$ in the transitions function's codomain is the set of all distributions over the set S. This is where the randomization arises.

Figure 2.1 shows an example MDP with states $S = \{S_0, S_1, S_2, S_3\}$, actions $A = \{a, b\}$, initial state S_0, and the transition function p given by the tables.

A policy σ for the MDP $\mathcal{M}$ is a function that decides what actions to take. We consider two types of policies: a *memoryless policy* and a *policy with memory*. A memoryless policy takes the form $\sigma \colon S \to Dist(A)$. The policy maps each state to a probability distribution over the actions. This means the action can be different if the same state is visited multiple times. However, the policy itself will not change when a state is visited again, the distribution stays the same. If the distribution does change when the state is visited again the policy is dependent on the path taken to get there. So the policy remembers the previously visited states and taken actions, i.e. the policy has memory. We define $Path$ to be the set of paths in the form $\pi = s_0 a_0 s_1 a_1 \dots s_n$ with $s_i \in S$ and $a_i \in A$. A policy with memory is then $\sigma \colon Path \to Dist(A)$.

Traversing the MDP using a policy will result in reaching each state with a certain probability [1]. We denote the probability of reaching a state $G \in S$ starting from state $s \in S$ using a strategy σ by $\mathbb{P}_{\mathcal{M}}(s \to G \,|\, \sigma)$. If $s = s_0$



	S_0	S_1	S_2	S_3
S_0	0.25	0.25	0.5	0
S_1	0	0.5	0.5	0
S_2	0	1	0	0
S_3	0	0	0	1

(a) Action a

	S_0	S_1	S_2	S_3
S_0	0	0	1	0
S_1	0	1	0	0
S_2	0.5	0	0	0.5
S_3	0	0	0	1

(b) Action b

Figure 2.1: Example MDP with transition function p.

the starting state we denote the chance as $\mathbb{P}_{\mathcal{M}}(G \,|\, \sigma)$. We say *almost-sure reachability* is achieved for a goal state G if $\mathbb{P}_{\mathcal{M}}(G \,|\, \sigma) = 1$. The policy σ is also called *winning* in this case.

A policy for the MDP in Figure 2.1 that achieves almost-sure reachability for the goal state $G = S_3$ would be: $\sigma(S_0) = \sigma(S_1) = \sigma(S_2) = b$. As the domain of σ is S this policy is memoryless.

Note that there is a path in the MDP that this winning policy σ can take that does not reach the goal state: $\pi = S_0 b S_2 b S_0 b S_2 \ldots$. However, the probability that the policy takes this path is $\lim_{n\to\infty}(0.5)^n = 0$, so the probability for reaching the goal state is still one. This is why the definition is called *almost*-sure reachability.

For simplicity sake in this thesis we assume that there is just one goal state G. Furthermore, we assume this state G is *absorbing*, meaning that all transitions leading from G lead back to G.

Note that we can do this without loss of generality: making the goal state absorbing does not change the probability that the state is reached, and

if there are multiple goal states, we can add an additional state G with transitions from all goal states to G.

Definition 2 (MEMDP)**.** A MEMDP is a tuple $\mathcal{M} = \langle S, A, S_0, \{p_i\}_{i \in I} \rangle$ where S, A, S_0 are the same as for an MDP, and $\{p_i\}_{i \in I}$ a finite set of transition functions, where I is a finite set of environment indices.

So a MEMDP is a set of MDPs that all share the same states, actions, and starting state. This way, we can define a policy without knowing in which environment we are acting. We reference a specific MDP (environment) using the notation $\mathcal{M}_i$ for $i \in I$. On a MEMDP we define almost-sure reachability achieve for a single policy the goal state is reached in all environments: for all $i \in I$, we have $\mathbb{P}_{\mathcal{M}_i}(G \,|\, \sigma) = 1$.

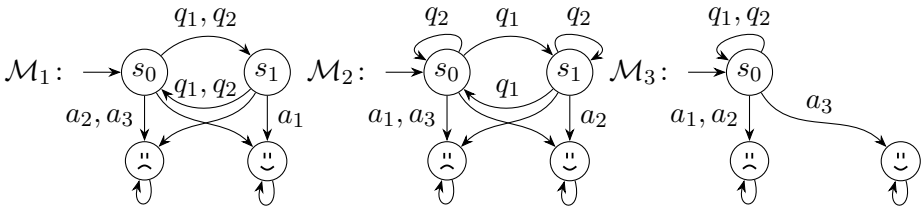


Figure 2.2: Example MEMDP, taken from [6].

Figure 2.2 shows an example MEMDP. The idea is to 'ask' questions q_1 and q_2, which yield responses of either 'switch' or 'stay'. By interpreting these responses, the agent can determine the specific environment it is operating in. Based on this knowledge, the agent then selects one of a_0, a_1, a_2 to 'answer' the question of what environment it is in, in an attempt to end up in ⍤. Note that to improve readability some shorthand notation is used: all probabilities in this MEMDP are zero or one so these are not explicitly shown, and the fact that ⍤ and ⍨ are absorbing is indicated by a looping arrow.

We exemplify a winning policy. This policy starts by 'asking' q_1. On switch the policy answers a_1, on stay it 'asks' q_2 and then on switch it answers a_2 and on stay a_3. Note that this policy takes memory, you need to consider both 'answers' before guessing the right environment. In fact, this MEMDP is not winnable with a memoryless policy. This example is worked out in more detail in Chapter 4.

Chapter 3

SAT Translation

In this chapter we translate the decision problem for finding a memoryless policy that almost-surely reaches a goal state G to SAT. This translation forms the basis for the translation to QBF in the next chapter.

The SAT translation and Theorem 1 which is required for it, are based on a paper by Chatterjee et al. [2] where they provide an analogous construction for Partially Observable Markov Decision Processes (POMDPs) [5].

Equivalent Statement Almost-Sure Reachability

Theorem 1 below is the basis for both translations. It gives an equivalent statement to almost-sure reachability that can be captured well by propositional formulas.

We define $R_\sigma \subseteq S$ to be the set of states reachable in an MDP with policy σ. For a MEMDP we define $R^i_\sigma \subseteq S$ to be the set of states reachable in the MDP $\mathcal{M}_i$ with policy σ. We define $\pi_k(s, s') = s_0 a_0 s_1 a_1 \dots a_{k-2} s_{k-1} \in Path$ to be a path of length k from $s_0 = s$ to $s_{k-1} = s'$. We say that $\pi_k(s, s')$ is *compatible* with the policy σ if $\sigma(s_i)(a_i) > 0$ for all $0 \leq i < k - 1$. In other words, the policy σ could have taken this path in the MDP.

Theorem 1. *A memoryless policy σ is winning in an MDP $\mathcal{M}$ if and only if for all states $s \in R_\sigma$ there is a path $\pi_k(s, g)$ from s to the goal state G that is compatible with σ and has length at most $k = |S|$.*

The proof is given in Appendix A.

3.1 Translation

Below we define, for a MEMDP $\mathcal{M}$ and a goal state G, the propositional formula $\phi_{\mathcal{M},G}$. After that, we prove in Theorem 2 that the decision problem

polynomially reduces to deciding if $\phi_{\mathcal{M},G} \in \text{SAT}$ and show how a satisfying valuation to ϕ relates to a policy in the MEMDP.

Boolean Variables

We introduce the following boolean variables:

- $\{A_{sa}\}$ for $s \in S$, $a \in A$, indicating that $\sigma(s)(a) > 0$. Note that it is not important what the exact value is of $\sigma(s)(a)$ is because we are considering almost-sure reachability.
- $\{S_{is}\}$ for $i \in I$, $s \in S$, indicating $s \in R^i_\sigma$.
- $\{P_{isk}\}$ for $i \in I$, $s \in S$, $k \in \mathbb{N}, 0 \leq k \leq |S|$, indicating that there is a path compatible with σ from state s to the goal state G in at most k steps.

Logical Constraints

The formula $\phi_{\mathcal{M},G}$ is defined as the conjunction of all constraints defined below.

In each state at least one action should be taken. For all $s \in S$:

$$\bigvee_{a \in A} A_{sa} \tag{3.1}$$

If a state s is reachable in an environment i and action a is taken in state s, then all states $t \in S$ with $p_i(s,a)(t) > 0$ are reachable, so S_{it} should be true. For all $i \in I$, $a \in A$, and $s, t \in S$ with $p_i(s,a)(t) > 0$:

$$(S_{is} \wedge A_{sa}) \rightarrow S_{it} \tag{3.2}$$

The initial state s_0 needs to be true because it is definitely reachable. For all $i \in I$:

$$S_{is_0} \tag{3.3}$$

If a state is reachable under σ in an environment, there should be a path from that state compatible with σ to the goal state G in $k = |S|$ steps. For all $i \in I$, and $s \in S$:

$$S_{is} \rightarrow P_{isk} \tag{3.4}$$

From the goal state the path to the goal state has length zero. This constraint is required because the P variables for other states flow from there with Constraint 3.7. For all $i \in I$, and $k \in \mathbb{N}$ with $0 \leq k \leq |S|$:

$$P_{iGk} \tag{3.5}$$

Only the goal state has a path of length zero to the goal state, any other state cannot reach the goal state with a path of length zero. For all $i \in I$, and $s \in S \setminus \{G\}$:

$$\neg P_{is_0} \tag{3.6}$$

The next constraint makes sure that the P variables are assigned correctly. It does this by ensuring that the variable P_{isk} can only be true, indicating there is a path of length at most k to G from s in environment i, if and only if from state s we can take some action a that can take us to a state t from which it is possible to reach the goal state in at most $k-1$ steps. For all $i \in I$, $s \in S$, and $k \in \mathbb{N}$ with $1 \leq k \leq |S|$:

$$P_{isk} \leftrightarrow \bigvee_{a \in A} \left(A_{sa} \wedge \left(\bigvee_{\substack{t \in S \\ p_i(s,a)(t)>0}} P_{it(k-1)} \right)\right) \tag{3.7}$$

Now that we have defined $\phi_{\mathcal{M},G}$, we can show that translation indeed polynomially reduces the decision problem.

Theorem 2. *The decision problem for finding a memoryless policy in the MEMDP $\mathcal{M}$ that almost-surely reaches the goal state G polynomially reduces to deciding if $\phi_{\mathcal{M},G} \in SAT$:*

- *For a MEMDP $\mathcal{M}$ and a goal state G, there exists a memoryless winning policy in $\mathcal{M}$ if and only if $\phi_{\mathcal{M},G} \in SAT$.*
- *$\phi_{\mathcal{M},G}$ can be computed in polynomial time.*

The proof is given in Appendix A.

Chapter 4

QBF Translation

In this chapter we provide the translation from the decision problem for finding a policy with memory in a MEMDP to a Quantified Boolean Formula (QBF formula). It is known that this translation had to exist because it was shown that the decision problem is PSPACE-complete [6].

We start by explaining how a winning policy with memory for the MEMDP in Figure 2.2 can be built up. In this example we show the core idea of the translation: a policy with memory can be seen as a set of memoryless policies, between which the policy switches when taking certain paths. We then show how we can extend this idea of layers of memoryless policies to a general form of a policy with memory. Finally, we provide the translation to the QBF formula and explain in more detail how it works.

4.1 Example

We will show how to construct a winning policy with memory for the MEMDP in Figure 2.2. For reference we show the MEMDP in this chapter as well, see Figure 4.1.

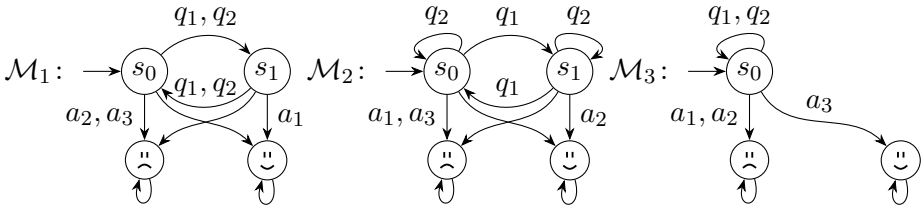


Figure 4.1: Copy of Figure 2.2.

The resulting policy with memory is illustrated in Figure 4.1. We start with the memoryless policy $\sigma_0 = \{s_0 \mapsto q_1, s_1 \mapsto q_2\}$. In state s_0 it takes action

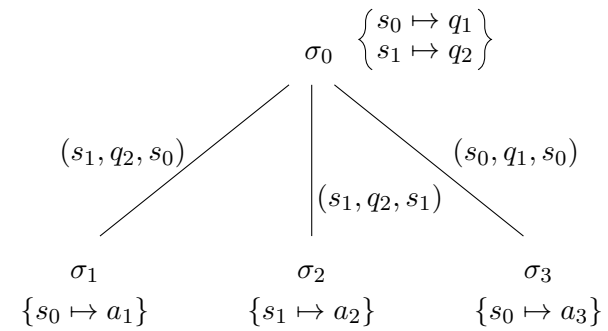


Figure 4.2: Illustration of a winning policy with memory for Figure 4.1.

q_1 and in state s_1 it takes action q_2. So we start in the initial state, s_0, and σ_0 prescribes we take action q_1. There are now two possibilities:

- If we are in environment one or two, we take the transition (s_0, q_1, s_1).
- If we are in environment three, we take the transition (s_0, q_1, s_0).

If we take transition (s_0, q_1, s_1), we know for certain that we are in environment three, because that transition only exists in that environment. So if we take this transition, we can switch to a different memoryless policy $\sigma_3 = \{s_0 \mapsto a_3\}$, that can act upon this newly gained information. In this scenario we know we are in environment three, so we can take action a_3 and reach the goal state. This is illustrated on the right in Figure 4.2. We call such a transition that reveals information about our current environment and causes us to switch to a new policy, a *frontier transition.*

Next we consider the other scenario where we take transition (s_0, q_1, s_1). We are now at state s_1 and σ_0 says we take action q_2. Again there are two possibilities:

- If we are in environment one, we take the transition (s_1, q_2, s_0).
- If we are in environment two, we take the transition (s_1, q_2, s_1).

Observe that both of these transitions reveal information about the current environment: if we take (s_1, q_2, s_0) we are certainly in environment one, and if we take transition (s_1, q_2, s_1) we are certainly in environment two because both of these transitions only exist in those environments. Thus, these transitions can be used as frontier transitions: if we take (s_1, q_2, s_0) we switch to the memoryless policy $\sigma_1 = \{s_0 \mapsto a_1\}$ and win, and if we take (s_1, q_2, s_1) we switch to the memoryless policy $\sigma_2 = \{s_1 \mapsto a_2\}$ and win. This again is illustrated in Figure 4.2.

So we have constructed a winning policy with memory for the MEMDP, by starting with the memoryless policy σ_0. Then, upon taking frontier

transitions that reveal what environment we are in exactly, we switch to a different memoryless policy (one of $\sigma_1, \sigma_2, \sigma_3$) that allows us to transition to the goal state.

4.2 General Form Policy with Memory

We extend this idea to a general form for policies with memory. Every policy with memory can be seen as memoryless policies between which the policy switches when taking frontier transitions.

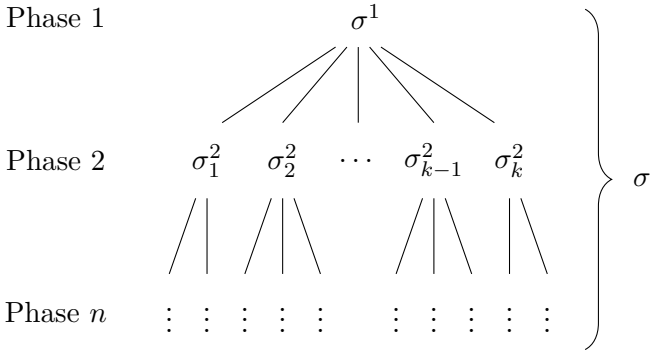


Figure 4.3: Illustration of how the memoryless policies σ_i^n (for $n \in I$) make up the policy with memory σ.

Figure 4.3 illustrates the general form of a policy with memory in a tree form. We start at the memoryless policy σ^1, and upon taking a frontier transition, we switch to a next memoryless policy σ_i^2, and upon taking another frontier transition we switch to some other memoryless policy σ_j^3. We call each layer of memoryless policies in the tree a *phase*.

Note that the number of children per node (subsequent policies a policy can switch to) may vary and depends on the number of frontier transitions. The depth of the tree also varies. It is possible that no frontier transitions need to be taken in the first phase, and almost-sure reachability to the goal state is already achieved in phase one. In this case, a memoryless strategy suffices. It is also possible that more phases are required for some paths than for others. In this case some branches in the tree might stop sooner than others. The maximum depth of the tree is $|I|$, the maximum number of phases, as follows from the following lemma:

Lemma 1. *If a policy can be found using more than $|I|$ phases in the QBF formula, a policy also exists with $|I|$ or fewer phases.*

The proof is given in Appendix A.

Corollary 1. *If no policy can be found using $|I|$ phases, no policy exists.*

Figure 4.3 also illustrates an important fact: the winning policy can be exponentially large. The QBF formula we define below can only be used to decide in PSPACE *whether* a winning policy exists. Obtaining the policy can take exponential space as the policy itself already could take up exponential space. This was shown by Van der Vegt et al. [6, Theorem 4].

4.3 Translation

Next, we give the translation to the QBF formula, building upon the ideas explained above.

We recall that a QBF formula is an extension of propositional formulas. It is again a propositional formula, but now every variable in the formula is bound by either an existential ($\exists$) or a universal ($\forall$) quantifier. For instance, $\forall x \forall y \exists z[(\neg x \vee z) \wedge (\neg y \vee \neg z)]$ is a QBF formula. If such a formula evaluates to true, the formula is in the language TQBF (True Quantified Boolean Formula). A true QBF formula is also called valid.

In each phase we try to find a memoryless strategy such that in all environments, for all reachable states there is a path of at most $|S|$ steps to the goal state, i.e., the policy is almost-sure winning, according to Theorem 1. This is encoded in the formula in the same way as in the previous chapter with roughly the same variables and constraints. The differences are that now these variables are bound by the existential quantifier, and they exist for each phase.

If it is not possible to find a policy that reaches the goal state almost-surely, a transition (s, a, t) can be 'marked' as a frontier transition. In that case, taking that transition is also considered winning in that phase. So instead of the target being just the goal state, the target is now a set containing the goal state, but possibly also frontier transitions.

In the next phase the universal quantifier considers all those frontier transitions and 'picks' one at a time. A new memoryless policy needs to be found for after those transitions have been taken. So if in phase n the transition (s, a, t) is chosen as a frontier transition, and the universal quantifier considers that transition, phase $n+1$ starts in state t in that case. However, we only need to consider those environments where this transition could have been taken, so this (hopefully) will give more information. Note that it is not a requirement that the frontier transition reveals information about the current environment, although this will generally be the case.

In the final phase transitions can no longer be marked as frontier transitions. It is forced that the policy in that phase almost-surely reaches the goal state. This is because of Corollary 1.

Quantified Variables

Next we give the quantified variables. As explained above, for finding the memoryless policies in each phase the same variables are used as for the memoryless translation. These variables are the A, S, and P variables.

The variables F and C variables are new in this translation. $F^n_{(s,a,t)}$ means transition (s,a,t) is 'marked' as a frontier transition in phase n. $C^n_{(s,a,t)}$ indicates we now consider this frontier transition in phase n.

For a MEMDP $\mathcal{M}$ and a goal state G we define the QBF formula $\Phi_{\mathcal{M},G}$. This formula has the following form

$$\Phi_{\mathcal{M},G} = \exists_1 \ldots \forall_2 \ldots \exists_2 \ldots \ \ldots \ \exists_{n-1} \ldots \forall_n \ldots \exists_n \ldots [\Phi'_{\mathcal{M},G}],$$

so first all of the quantifiers, and then the propositional formula $\Phi'_{\mathcal{M},G}$ containing the quantified variables. In Theorem 3 we prove that the decision problem polynomially reduces to deciding if $\Phi_{\mathcal{M},G} \in$ TQBF and how a witness for the formula relates to a winning policy in the MEMDP.

By $(s,a,t) \in p$ we mean there is an $i \in I$ with $p_i(s,a)(t) > 0$. In other words: the transition exists in an environment.

$$\underbrace{\exists_1 \overbrace{A^1_{sa}}^{\substack{\forall s\in S\\ \forall a\in A}} \overbrace{S^1_{is}}^{\substack{\forall i\in I\\ \forall s\in S}} \overbrace{P^1_{isk}}^{\substack{\forall i\in I\\ \forall s\in S\\ \forall k\in\mathbb{N}\ s.t.\ 0\le k\le |S|}} \overbrace{F^1_{(s,a,t)}}^{\forall (s,a,t)\in p}}_{\text{Phase 1}}$$

$$\underbrace{\forall_2 \overbrace{C^2_{(s,a,t)}}^{\forall (s,a,t)\in p} \exists_2 \overbrace{A^2_{sa}}^{\substack{\forall s\in S\\ \forall a\in A}} \overbrace{S^2_{is}}^{\substack{\forall i\in I\\ \forall s\in S}} \overbrace{P^2_{isk}}^{\substack{\forall i\in I\\ \forall s\in S\\ \forall k\in\mathbb{N}\ s.t.\ 0\le k\le |S|}} \overbrace{F^2_{(s,a,t)}}^{\forall (s,a,t)\in p}}_{\text{Phase 2}}$$

$$\vdots$$

$$\underbrace{\forall_{|I|-1} \overbrace{C^{|I|-1}_{(s,a,t)}}^{\forall (s,a,t)\in p} \exists_{|I|-1} \overbrace{A^{|I|-1}_{sa}}^{\substack{\forall s\in S\\ \forall a\in A}} \overbrace{S^{|I|-1}_{is}}^{\substack{\forall i\in I\\ \forall s\in S}} \overbrace{P^{|I|-1}_{isk}}^{\substack{\forall i\in I\\ \forall s\in S\\ \forall k\in\mathbb{N}\ s.t.\ 0\le k\le |S|}} \overbrace{F^{|I|-1}_{(s,a,t)}}^{\forall (s,a,t)\in p}}_{\text{Phase } |I|-1}$$

$$\underbrace{\forall_{|I|} \overbrace{C^{|I|}_{(s,a,t)}}^{\forall (s,a,t)\in p} \exists_{|I|} \overbrace{A^{|I|}_{sa}}^{\substack{\forall s\in S\\ \forall a\in A}} \overbrace{S^{|I|}_{is}}^{\substack{\forall i\in I\\ \forall s\in S}} \overbrace{P^{|I|}_{isk}}^{\substack{\forall i\in I\\ \forall s\in S\\ \forall k\in\mathbb{N}\ s.t.;0\le k\le |S|}}}_{\text{Phase } |I|}$$

Logical Constraints

Next we look at the the formula $\Phi'_{\mathcal{M},G}$ which depends on these variables. The formula $\Phi'_{\mathcal{M},G}$ is defined as the conjunction of all constraints defined below, for all phases $n \in I$:

In each state at least one action should be taken. For all $s \in S$:

$$\bigvee_{a \in A} A^n_{sa} \tag{4.1}$$

If s is reachable in environment i and action a is taken, the states reachable with that action are now also reachable. If the transition is chosen as a frontier transition, t does not have to be considered reachable in this phase. For all $i \in I$, $a \in A$, and $s, t \in S$ with $p_i(s, a)(t) > 0$:

$$(S^n_{is} \wedge A^n_{sa}) \rightarrow (S^n_{it} \vee F^n_{(s,a,t)}) \tag{4.2}$$

If σ can reach state s in an environment, there should be a path from that state conforming to σ to the target set for this memoryless policy in $k = |S|$ steps. For all $i \in I$, and $s \in S$:

$$S^n_{is} \rightarrow P^n_{isk} \tag{4.3}$$

From the goal state the path to the goal state has length zero. This constraint is required because the P variables for other states flow from here with constraint 4.6. For all $i \in I$, and $k \in \mathbb{N}$ with $0 \leq k \leq |S|$:

$$P^n_{iGk} \tag{4.4}$$

Only the goal state has a path of length zero to the goal state, any other state cannot reach the goal state with a path of length zero. For all $i \in I$, and $s \in S \setminus \{G\}$:

$$\neg P^n_{is0} \tag{4.5}$$

The next constraint makes sure the P variables are assigned correctly. It does this by ensuring that the variable P^n_{isk} can only be true, indicating there is a path of length at most k to the goal set from s in environment i, if and only if from a state s we can take some action a that can take us to a state t from which it is possible to reach the goal set it at most $k - 1$ steps, or, if it leads to a frontier transition. This way for a frontier transition (s, a, t) the literals $P^n_{is1}, P^n_{is2}, \ldots, P^n_{is|S|}$ are true, indicating the goal can be

reached in $1, 2, \ldots, |S|$ steps from s, which in this phase means taking the frontier transition. For all $i \in I$, $s \in S$, and $k \in \mathbb{N}$ with $1 \leq k \leq |S|$:

$$P^n_{isk} \leftrightarrow \bigvee_{a \in A} \left(A^n_{sa} \wedge \left(\bigvee_{\substack{t \in S \\ p_i(s,a)(t) > 0}} (P^n_{it(k-1)} \vee F^n_{(s,a,t)}) \right) \right) \tag{4.6}$$

In phase one the starting state is just the starting state of the MEMDP s_0. For all $i \in I$:

$$S^1_{is_0} \tag{4.7}$$

For subsequent phases, the starting states depend on the frontier transitions picked in the last phase. If $C^n_{(s,a,t)}$ is true and $F^{n-1}_{(s,a,t)}$ is too, meaning that (s, a, t) was chosen as a frontier state in the last phase, we consider all environments in which that transitions exists and was taken. For all $i \in I$, $a \in A$, and $s, t \in S$ with $p_i(s, a)(t) > 0$:

$$(F^{n-1}_{(s,a,t)} \wedge C^n_{(s,a,t)} \wedge S^{n-1}_{is} \wedge A^{n-1}_{sa}) \rightarrow S^n_{it} \tag{4.8}$$

Now that we have defined $\Phi_{\mathcal{M},G}$, we can show that the translation indeed polynomially reduces the decision problem.

Theorem 3. *The decision problem for finding a policy in the MEMDP $\mathcal{M}$ that almost-surely reaches the goal state G polynomially reduces to deciding whether $\Phi_{\mathcal{M},G} \in$ TQBF:*

- *For a MEMDP $\mathcal{M}$ and a goal state G, there exists a winning policy in $\mathcal{M}$ if and only if $\Phi_{\mathcal{M},G} \in$ TQBF*
- *$\Phi_{\mathcal{M},G}$ can be computed in polynomial time.*

The proof is given in Appendix A.

Chapter 5

Experimental Results

We implemented the translation to a QBF formula to test its application for deciding the decision problem. The implementation converts a MEMDP to the corresponding QBF formula and then uses the Z3 theorem solver from Microsoft [3] to determine its validity, thereby establishing the truth value of the decision problem.

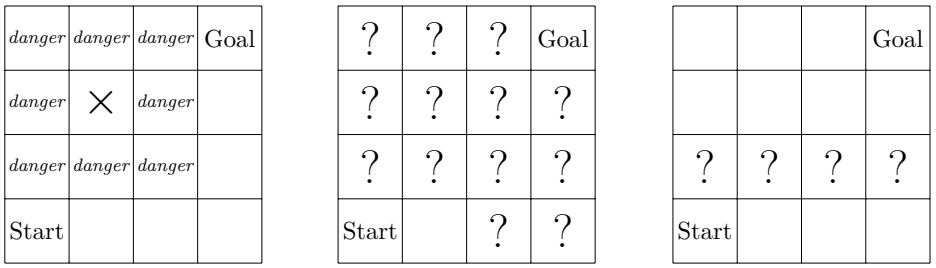
Technical Specifications The experiments were carried out on a 2.3 GHz Dual-Core i5 processor. Due to small differences between identical experiments, all experiments were done three times and the times in the tables are the average of those results. The provided time is the time it took Z3 to prove (in)validity after it was given the formula. The time it takes for the MEMDP to be converted to the formula is not included because it runs in polynomial time and we used a simple implementation that can easily be improved. We used a 30 minute time limit for all experiments. The source code of the implementation is available on GitHub[1].

5.1 Models Tested

Grid

The first models we tested represent a set of games, called Grid, where the agent needs to traverse a two-dimensional grid of $n \times n$ cells to reach a goal. An illustration of the game can be seen in Figure 5.1. The agent starts in the bottom left cell and needs to reach the top right by moving to the north, east, south, or west. Somewhere on the grid there is a hole that the agent needs to avoid (shown in Figure 5.1a by $\times$). If the agent moves into the cell where the hole is located, it is stuck and cannot reach the goal anymore.

[1] `https://github.com/Jorritboer/memdp-qbf`



(a) Example of an environment in Grid4.

(b) Possible hole locations in Grid4.

(c) Possible hole locations in NGrid4.

Figure 5.1: Illustration of the Grid models.

In the cells around the hole, the agent can 'detect' that the hole is close. This is illustrated in Figure 5.1a by *danger*. Using this information it can traverse around the hole. Each environment represents a possible location for the hole. Figure 5.1b shows the possible locations for the 4×4 grid. Note that in the cell to the right of the start, there cannot be a hole, as it would prevent any winning policy due to the possibility of encountering the hole on the first move. We denote the specific model by the size of the grid, for example Grid5 is the model for a grid of 5×5 cells.

For testing the not valid models we use a variant of Grid, which we refer to as NGrid, where the agent cannot detect that the hole is close. Due to this change the agent cannot know where the hole is and thus not guarantee winning in all environments. For NGrid also restrict the possible locations for the hole to one row, because otherwise the time taken to prove invalidity already goes past the timeout for the first model. An example is shown in Figure 5.1c.

Mastermind

The second set of MEMDP models we tested are simplified versions of the game Mastermind. In Mastermind a secret code consisting of a number of balls of different colors is given. The player's objective is to guess this code within a limited number of turns. After each guess, the player receives feedback indicating the number of colored balls in the correct positions. In the MEMDP representation of the game, each environment represents a different possible secret code, and the actions represent guesses.

Different versions of the game are determined by three variables: the number of possible colors, the number of guesses the player gets, and the number of balls that the code consists of. We denote each version of the game by these variables. For instance, M243 has two colors, four guesses, and three balls.

5.2 Results

Invalid Models

We first provide some results for models where the decision problem is false, i.e. the game is not always winnable with the provided number of guesses. The corresponding QBF formula will thus be invalid. Table 5.1 shows the invalid models and the time it took for Z3 to prove they were false. The table also includes information about the MEMDP and the size of the resulting QBF formula, Z3 provides the option to write the formula to a text file, this gives us a measure for the size of the formula. As expected from Theorem 3, this is $\mathcal{O}(|S|^3 \cdot |A| \cdot |I|^2)$.

Model	$\|I\|$	$\|S\|$	$\|A\|$	$\|QBF\|$	Time (s)
M221	2	3	2	10 KB	0.041
M222	4	7	4	184 KB	1.8
M233	8	13	8	27.6 MB	Timeout
NGrid3	3	9	4	298 KB	2
NGrid4	4	16	4	2.3 MB	Timeout

Table 5.1: Experimental Results For Invalid QBF Formulas

The data shows that the implementation is not a very effective way of proving that no winning policy exists. The time required to prove invalidity is significantly larger compared to the results obtained by Van der Vegt et al. [6], who developed a specific algorithm. The main factor for this is the number of phases, because each phase comes with universally quantified variables, which have a big impact on the complexity of the problem. As per Corollary 1 we can only prove that no policy can be found if the formula with $|I|$ phases is invalid, so for proving invalidity all phases will have to be used.

Valid Models

Proving a policy does exist can be easier because it is possible less phases are required. In fact, for the tested models only one or two phases were required. For each model we first tested whether one phase was sufficient, and if it was not, we used two. Table 5.2 provides the results for valid formulas, including the number of phases used. Note that when the number of phases is one the formula is effectively the SAT formula.

Model	$\|I\|$	$\|S\|$	$\|A\|$	$\|QBF\|$	# Phases	Time (s)
M221	2	5	2	21 KB	1	0.019
M232	4	10	4	227 KB	1	0.038
M243	8	17	8	2.2 MB	1	0.29
M342	9	13	9	1.9 MB	2	0.47
M452	16	16	16	8.4 MB	2	4.7
M353	27	21	27	36.6 MB	2	620
M364	81	31	81	N/A[2]	2	Timeout
Grid3	6	17	4	1.1 MB	2	1.6
Grid4	13	32	4	8 MB	2	5.6
Grid5	22	50	4	33.1 MB	2	26
Grid6	33	72	4	103.1 MB	2	300
Grid7	47	98	4	N/A[2]	2	Timeout

Table 5.2: Experimental results for valid QBF formulas.

The data is plotted in Figure 5.2. As you can see, both lines seem to follow a linear trajectory, although at a different gradient. Due to the logarithmic y-axis, the linear trajectories indicate an exponential time requirement as a function of the size of the QBF formula.

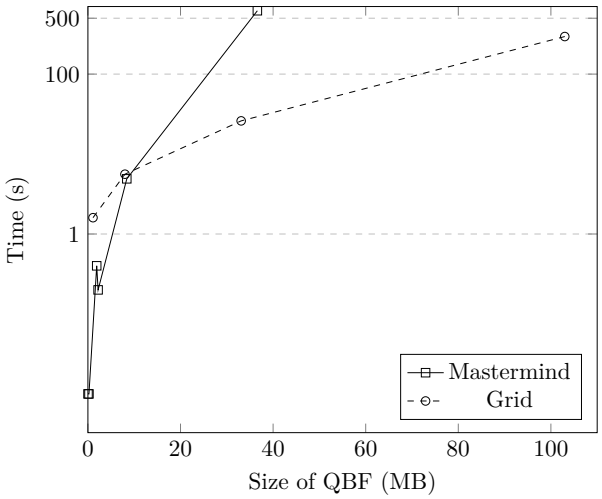


Figure 5.2: Graph of the data in Table 5.2.

We are unsure why the two lines for the different models seem to exhibit a different gradient. It is possible that the number of actions, which increase for Mastermind but not for Grid, has a big impact on the complexity of the resulting formula. It is also possible that the Grid models require a less complicated policy with fewer frontier transitions.

[2]Our machine was not able to write the QBF formula to a text file for these models.

The times required to prove validity, i.e., that a policy does exist for the game, are more competitive, although still not as fast as those obtained by Van der Vegt et al. [6].

It is worth mentioning that when only one phase is required, effectively reducing the formula to a SAT formula, the winning memoryless strategy can be extracted. Because there are no universal quantifiers, Z3 can output a satisfying valuation, from which we can construct the strategy. This is not possible for more than one phase because the policies in the following phases depend on the universal quantifier.

Chapter 6

Discussion on Reducing the Number of Phases

We showed in the previous chapter that the required number of phases is often a lot lower than $|I|$. This observation raises the question whether the upper limit can be reduced. Especially considering the number of phases appears to be the primary factor contributing to the slowdown of the theorem solver. By reducing the number of phases, we can potentially significantly improve the efficiency of the provided approach.

Although we did not obtain definitive results in achieving this goal, we explored some promising approaches. In this chapter, we will delve into these approaches, laying the groundwork for potential future research.

Features

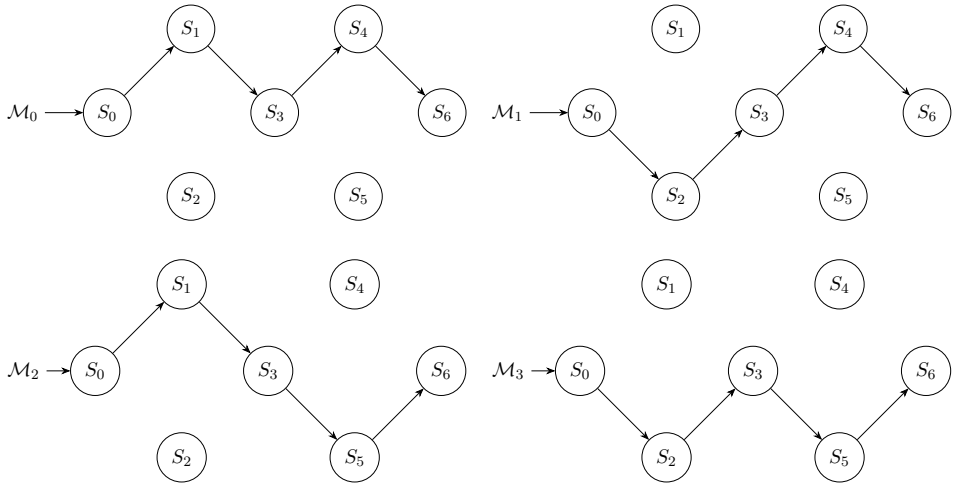
Our approach relies on describing the set of environments in terms of a set of *features* and define I as all possible combinations of these features. For instance, consider again the Grid model of the previous chapter. A hole is located on a two-dimensional grid, so we can define its position by two features: the x and y position. If we can now gain information about these features, we can gain a lot of information about what environment we are in. For example, if we learn the hole is not on the row for $x = n$, we can eliminate a possible number of environments equal to y.

We define the feature set $F = \{F_0, F_1, \ldots, F_n\}$, where $F_i \in \{0, 1\}$. For now, we restrict our consideration to binary features. This enables us to formulate propositional formulas using these features. We now redefine the environment set as all possible combinations of these features. This means that for n features, we have 2^n environments. The key to this approach lies in the way we construct the transition function p. For each feature $F_i \in F$

we define a set of transitions $T_{F_i} = \{(s_0, a_0, t_0), (s_1, a_1, t_1), \ldots, (s_k, a_k, t_k)\}$ that are present in an environment if $F_i = 1$ and another set $T_{\neg F_i} = \{(s_0, a_0, t_0), (s_1, a_1, t_1), \ldots, (s_k, a_k, t_k)\}$ that are present in an environment if $F_i = 0$. The transition function is now restriced as follows:

$$\{(s, a, t) \mid (s, a, t) \in p\} = \bigcup_{F_i \in F} (T_{F_i} \cup T_{\neg F_i})$$

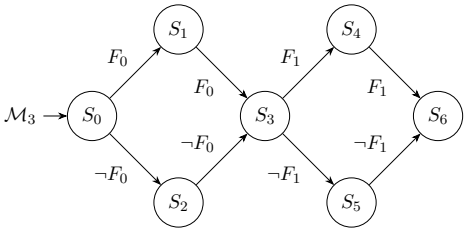
Note that these sets of transitions need not be disjoint, they can, and often will, contain the same transitions. Our interest lies in transitions that occur in some environments but do not in others. These will be frontier transitions in the policy with memory and make the policy gain information about which environment it is in. In this setup, such a transition will necessarily exist in half of the environments, thus allowing the elimination of half of the possible environments.



(a) MEMDP written out.

	F_0	$\neg F_0$
F_1	0	1
$\neg F_1$	2	3

(b) Table showing which features describe what environment.



(c) MEMDP described using features.

Figure 6.1: MEMDP explicitly written out, and denoted using features.

We illustrate with an example. Figure 6.1 presents a MEMDP with four environments, represented in two different ways. Firstly, in Figure 6.1a the

environments are explicitly written out, and secondly, in Figure 6.1c they are described using the features $F = \{F_0, F_1\}$.

Consider the problem of traversing the MEMDP and in state S_6 answering in which environment we are in in order to reach the goal state. The QBF translation of this MEMDP would require $|I| = 4$ phases to solve. However, if we consider the MEMDP as described by the features, we only need $|F| = 2$ phases. Consider a policy taking the transition (S_0, S_1). This means $F_0 = 1$, which implies that we can eliminate environments 1 and 3. Using this approach of finding the values of the features to establish the environment we can demonstrate that we only need a number of phases equal to the number of features $|F|$, which will be logarithmic in the number of environments.

However, there is a drawback to this approach. While it decreases the number of phases, the constraints on the transition function reduces the expressiveness of MEMDPs. We found that this decrease in expressiveness was too significant to remain useful, as it prevented modeling most examples. For example, we were unable to model the Grid and Mastermind models from the previous chapter using binary features.

Increasing Expressiveness

The goal would be to increase expressiveness of the MEMDPs while keeping the number of phases equal to the number of features. One approach would be to extend the idea explained above of binary features to features that can take on more than two values.

Another possible approach is to not just have sets of transitions per truth value of each feature, but to define propositional formulas depending on these features to construct the sets of transitions. More concretely: a set of features T_ϕ that exist in an environment if the formula $\phi(F_0, F_1, \ldots, F_n)$ containing $F_0, F_1, \ldots F_n$ evaluates to true.

In a MEMDP set up this way, a path through a set of frontier transitions would provide a set of formulas $\Gamma = \{\phi_0, \phi_1, \ldots, \phi_n\}$ that we know are true. If we can somehow show, or modify the constraints to be able to show, that for $|F|$ transitions, this set of formulas $\Gamma = \{\phi_0, \phi_1, \ldots, \phi_n\}$ provides a unique valuation for the features, then we would achieve our objective.

Chapter 7

Conclusions

In this thesis, we consider MEMDPs and finding a policy that achieves almost-sure reachability. We provide a translation from a MEMDP to a propositional formula that is satisfiable if and only if there exists a memoryless policy that achieves almost-sure reachability. We build upon this SAT translation to make a translation to a QBF formula that is valid if and only if a winning policy with memory exists. This translation is based upon the idea of 'phases' of memoryless policies, between which the policy with memory switches when taking selected transitions.

The translation to a QBF formula was implemented and tested using the Z3 theorem prover. This approach to deciding whether a policy exists did not yield competitive results in comparison to existing strategies.

Finally, we provided a discussion on decreasing the number of phases required for the QBF formula. This could significantly increase the efficiency of the approach.

Future work could focus on refining and concretizing the ideas suggested for reducing the number of phases. Additional ways the approach could be improved include improving the QBF formula by reducing the number of variables and constraints, considering alternative theorem solvers, and employing different solving tactics. It is also worth investigating whether the limitations encountered during testing were specific to the models used, and whether the approach may prove more effective for different types of models.

Bibliography

[1] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.

[2] Krishnendu Chatterjee, Martin Chmelik, and Jessica Davies. A symbolic SAT-based algorithm for almost-sure reachability with small strategies in POMDPs. In *AAAI*, pages 3225–3232. AAAI Press, 2016.

[3] Leonardo Mendonça de Moura and Nikolaj S. Bjørner. Z3: an efficient SMT solver. In *TACAS*, volume 4963 of *Lecture Notes in Computer Science*, pages 337–340. Springer, 2008.

[4] Jean-François Raskin and Ocan Sankur. Multiple-environment markov decision processes. In *FSTTCS*, volume 29 of *LIPIcs*, pages 531–543. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014.

[5] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach (4th Edition)*. Pearson, 2020.

[6] Marck van der Vegt, Nils Jansen, and Sebastian Junges. Robust almost-sure reachability in multi-environment MDPs. In *TACAS (1)*, volume 13993 of *Lecture Notes in Computer Science*, pages 508–526. Springer, 2023.

Chapter 8

Appendix

Proof of Theorem 1.

$\Rightarrow$ Assume σ is a memoryless winning strategy. Then, for all $s \in R_\sigma$ we know $\mathbb{P}_{\mathcal{M}}(s \to G \,|\, \sigma) = 1$. So there must be a path from s to the goal state G. The shortest path from s to the goal state must have length $j \leq k$. Because if it is not, so $j > k = |S|$, that means we have visited a state twice, which contradicts our assumption that we have the shortest.

$\Leftarrow$ Now assume σ is a memoryless strategy such that for all $s \in R_\sigma$ there is a path $\pi_j(s, G)$ of length at most k, $0 \leq j \leq k$. Start in the starting state s_0. Take k steps from s_0. Because there is a finite path of length at most k to G from s_0, the probability that we will have reached the goal state is bigger than 0. Take p_0 as the chance this does not happen. Note $0 \leq p_0 < 1$. If it does not happen we are in another state $s_1 \in R_\sigma$. So again we can take k steps and have a probability p_1 that we do not reach the goal state. Again $0 \leq p_1 < 1$. Let $p_{\max}$ be the maximum chance of not reaching the goal state G in MDP. The chance that we will never reach the goal state is $p_0 \cdot p_1 \cdot \cdots \cdot p_{n-1} \cdot p_n \leq (p_{\max})^n = 0$ for $n \to \infty$. So $\mathbb{P}_{\mathcal{M}}(G \,|\, \sigma) = 1$. $\square$

Proof of Theorem 2.

$\Rightarrow$ Assume there is a memoryless policy $\sigma \colon S \to \mathit{Dist}(A)$ that almost-surely reaches the goal state G. We give a valuation $v \colon \text{Atoms} \to \{0, 1\}$ such that $v(\phi_{\mathcal{M},G}) = 1$.

We assign the following variables to true and all others to false.

For all $s \in S, a \in A, i \in I$ and $k \in \mathbb{N}$ with $0 \leq k \leq |I|$:

$$\begin{aligned} v(A_{sa}) = 1 &\leftrightarrow \sigma(s)(a) > 0 \\ v(S_{is}) = 1 &\leftrightarrow s \in R^i_\sigma \\ v(P_{isk}) = 1 &\leftrightarrow \text{there is a path } \pi_j(s, G) \text{ for some} \\ &\quad\ 0 \leq j \leq k \text{ compatible with } \sigma \end{aligned}$$

- Constraint 3.1 is satisfied because for all reachable states we can assume the policy has at least one action it can take, and for unreachable states we can randomly pick an A variable to assign true.
- Constraint 3.2 is satisfied because of the definition of R^i_σ.
- Constraint 3.3 is satisfied because $s_0 \in R^i_\sigma$.
- Constraint 3.4 is satisfied because σ almost-surely reaches G and Theorem 1.
- Constraint 3.5 holds because clearly for the goal state there is always a path of at most k to itself, the empty path.
- Constraint 3.6 holds because a path of 0 to the goal state can only be possible for the goal state itself.
- Constraint 3.7 is satisfied because if a path $\pi_k(s, G)$ exists, the policy has to have a path of length one to a state t from which a path $\pi_{k-1}(t, G)$ exists. Observe in particular that because the goal state G is absorbing it always has a path of length one to a state with a path of $k-1$ steps to the goal state, namely itself.

So $v(\phi_{\mathcal{M},G}) = 1$.

$\Leftarrow$ Assume $\phi_{\mathcal{M},G} \in$ SAT. This means that $\phi_{\mathcal{M},G}$ is satisfiable, and there exists a valuation $v\colon \text{Atoms} \to \{0, 1\}$ that assigns truth values to the boolean variables in ϕ such that $v(\phi_{\mathcal{M},G}) = 1$.

We define the policy $\sigma\colon S \to Dist(A)$ as follows: for each state s and action a if $v(A_{sa}) = 1$ then we set $\sigma(s)(a) > 0$. Because of Constraint 3.1 the policy will have at least one action it can take in each state.

Because of Constraint 3.3 it has to be the case that $v(S_{is_0}) = 1$ for all $i \in I$. Now from Constraint 3.2 it follows that for a transition (s, a, t) in environment $i \in I$, if s is reachable for the policy σ in environment i and the policy σ can take action a in state s ($\sigma(s)(a) > 0$), t will now be reachable in environment i. So for an environment i, the S variables will reflect whether σ can reach a state: $s \in R^i_\sigma \to v(S_{is}) = 1$.

From Constraint 3.4 it follows that if $v(S_{is}) = 1$ (state s is reachable for σ in environment i) the corresponding P variable must also be true for $k = |S|$:

$v(P_{isk}) = 1$. Constraint 3.6 makes sure this can only be the case if the policy can reach another state s_1 in one step that has a path compatible with σ to the goal state in $k-1$ steps, i.e. $v(P_{is_1(k-1)}) = 1$. Continuing this allows us to construct a path $\pi = sa_0s_1a_1s_2 \dots a_ks$ compatible with σ with $v(P_{is_{|S|-k}k}) = 1$ for $0 \leq k \leq |S|$. Now note that $v(P_{is_k0}) = 1$ can only be the case for $s_k = G$ the goal state because Constraint 3.5 makes sure $v(P_{iG0}) = 1$ and Constraint 3.6 forces $v(P_{it0}) = 1$ for $t \neq s_0$. Note that it is possible that the last states in the path are all the goal state, because G is absorbing.

So for the constructed policy σ, for all environments $i \in I$ and states $s \in R^i_\sigma$ reachable for σ there is a path of at length most $|S|$ to the goal state. From Theorem 1 it now follows that σ almost-surely reaches the goal state G.

Finally, the amount of clauses $\phi_{\mathcal{M},G}$ is $\mathcal{O}(|I| \cdot |A| \cdot |S|^3)$ due to Constraint 3.7, so $\phi_{\mathcal{M},G}$ can be computed in polynomial time.

□

Proof of Lemma 1.

The minimum amount of information gained from one phase to the next is ruling out one environment. Additionally, information gain in a MEMPD is monotonic: once we know that we are not in a particular environment, we never lose this knowledge [6].

So if a winning policy exists with more than $|I|$ phases, that policy transitioned between one memoryless strategy σ_1 to another σ_2 without gaining information about the environment it is in. We show that we can combine these policies into a single memoryless policy.

Say σ_1 almost-surely reaches some set G_1 which can include the goal state, and frontier transitions. G_1 at least contains the frontier transition (s, a, t). If that transition is taken the policy switches to σ_2. But, the set of environments $I' \subseteq I$ σ_1 and σ_2 'know' they can be in remains the same. σ_2 then starts at state t and almost-surely reaches a set G_2 which can also include the goal state and frontier transitions. We compute the set of states $R_{\sigma_2} = \cup_{i \in I'} R^i_{\sigma_2} \subseteq S$ that σ_2 can reach starting at state t while being in one of the environments in I'. Now define a new memoryless policy σ_1' which is the same as σ_2 in R_{σ_2} and the same as σ_1 in $S \setminus R_{\sigma_2}$. That is, for $s' \in R_{\sigma_2}$ $\sigma_1'(s') = \sigma_2(s')$ and for $s' \in S \setminus R_{\sigma_2}$ we have $\sigma_1'(s') = \sigma_1(s')$.

This new memoryless policy σ_1' will either follow σ_1 and reach G_1, or it will reach some state in R_{σ_2} (which includes t) and then operate like σ_2 did in its phase and almost-surely reach G_2. So it will almost-surely reach $G = G_1 \cup G_2$, i.e. the same result is reached with just one memoryless policy as was previously with two.

Note that this is only possible because the set of environments the two policies can be in are the same. Because of that, σ_2 cannot operate on different assumptions than σ_1, so its strategy will also work in the previous phase.

So we must gain information from every transition to the next phase. So a policy exists with $|I|$ phases or fewer. □

Proof of Theorem 3.

$\Rightarrow$ Assume there exists a policy with memory $\sigma\colon Path \to Dist(A)$ that almost-surely reaches the goal state G.

We turn this policy into a tree of memoryless policies, like in Figure 4.3. We define the starting policy by $\sigma^1(s_0) = \sigma(s_0)$, the action σ takes at the first state. Now for a path $\pi = s_0 a_0 s_1$ we switch from policy σ^1 to σ^2 if transition (s_0, a_0, s_1) is taken. And σ^2 is defined by $\sigma^2(s_1) = \sigma(\pi)$. We continue this construction to form the tree of memoryless strategies.

Because of Lemma 1 we can reduce this tree of memoryless policies to a tree with a maximum depth $|I|$. Now for every memoryless policy we can assign a valuation for the corresponding phase in the QBF formula, analogously to the $\Rightarrow$ proof of Theorem 2. Combining these valuations into a witness for the QBF formula is a valid witness.

$\Leftarrow$ Assume that $\Phi_{\mathcal{M},G} \in$ TQBF, so $\Phi_{\mathcal{M},G} \in$ is valid. This means there exists a witness for the QBF formula.

Because the first phase consists just of existentially quantified variables, we can get a valuation $v\colon$ Atoms $\to \{0, 1\}$ for these variables such that the clauses in $\Phi'_{\mathcal{M},G}$ that contain just these variables are true.

Analogously to the $\Leftarrow$ part of the proof of Theorem 2 we can show this assignment allows us to construct a memoryless policy σ^1 that almost-surely reaches some goal. The only difference being what this goal is. Instead of the goal being just the goal state, here we allow it to be a set, possibly containing transitions: $G_1 = \{G, T_0, T_1, \dots\}$. Where $T_i = (s_i, a_i, t_i) \in p$ a transition. If $v(F^1_{(s_i,a_i,t_i)}) = 1$ a transition is picked as a frontier transition, and 'added' to the goal set. This has an effect on Constraints 4.2 and 4.6. In Constraint 4.6 taking that transition is now considered winning, so the state s_i now has a path to the goal in one step. So $v(P^1_{js1}) = v(P^1_{js2}) = \cdots = v(P^1_{js|S|}) = 1$ for $j \in I$. And Constraint 4.2 allows $v(S^1_{js})$ to be false because we do not need to consider s to be visitable in this phase if it is reached through the transition (s, a, t). So σ_1 almost-surely reaches the goal set $G_1 = \{G, T_0, T_1, \dots\}$.

Now consider the next phase. For all frontier transitions T_i there will be a phase two where a memoryless policy σ_i^2 is found. For frontier transition $T_i = (s_i, a_i, t_i)$, per Constraint 4.8, t_i will be the starting state, but only in those environments the previous policy σ^1 could actually reach this state through T_i. These memoryless policies will again reach some goal set G_2. If this set contains transitions, a next phase will be considered. Until the final phase, where it is forced that $G_{|I|} = \{G\}$ is just the goal state.

Now assemble the policy with memory σ as follows: start with σ^1 and upon taking frontier transition T_i switch to policy σ_i^2, and again for transition T_j and σ_j^3, etc. This policy with memory will almost-surely reach the goal state G because every memoryless policy σ_i^n will almost-surely reach a frontier transition and switch to a next memoryless policy σ_j^{n+1}, or it will almost-surely reach the goal state G.

There are $\mathcal{O}(|S|^3 \cdot |A| \cdot |I|)$ clauses in Constraint 4.6 and all constraints are in I phases so there are $\mathcal{O}(|S|^3 \cdot |A| \cdot |I|^2)$ clauses. So computing $\Phi_{\mathcal{M},G}$ can be done in $\mathcal{O}(|S|^3 \cdot |A| \cdot |I|^2)$ time. □